Behavioral Context Recognition: a First Online Approach

Marika Pia Salvato, Muhammad Waseem

September 16, 2021

1 Introduction

Behavioral context recognition refers to inferring everything related to what a person is, what he's doing, who they are with, his location, and so on. The analysis of all these aspects is crucial for several applications. More so, *automatic* behavioral context recognition can make technologies more ubiquitous and unobtrusive so that they can act smarter according to the user's actions and environment and without user intervention.

A behavioral context-aware system that adapts its behavior according to our context could serve many applications. A music recommendation system could benefit from insight into the user's mood or activities and thus recommend more upbeat or relaxing music. Automatic behavior detection could be beneficial in healthcare, too: monitoring physical activity, mental states, living environment, etc. will help detect critical times and offer immediate support. Generally, all disciplines of life can benefit from the recognition of human actions and context, including patient activity monitoring for health care, children's activity tracking for their safety, and monitoring of the instructor's and students' activities in a lecture hall for improved learning. Automatic detection of human activity and context could be easily exploited by integration with smartphones and IoT devices: for example, when a user enters a museum, the user's smartphone senses that the current location is a museum from the GPS and surrounding building sensors. In order not to harm the people around user, user's smartphone automatically switches to silent mode[1].

Existing commercial tools are not sufficient since they offer superficial recognition of human activities (HAR), not taking into consideration the *context* and so sometimes failing to detect a specific action: for example, the sitting pattern may be different if a person is in a meeting, or in a car. Similarly, the gait pattern of a person may differ when the person is walking alone or walking with a group of friends [2]. Most of the literacy regarded the problem of automatic behavioral context recognition by processing data streams originating from different sensing modalities.

Vaizman et al.[3] presented an in-the-wild recognition model for human context, which utilizes the fusion of smartphone sensors (including accelerometer, gyroscope, magnetometer, a location sensor, microphone, and phone state) and watch accelerometer for the recognition of human behavioral context. They used a logistic regression classifier for recognizing multiple single-label contexts for daily living, which include a person's physical activity, phone position, and location.

Others like Saeed et al.[4] and Ehatisham-Ul-Haq et al.[5] used other machine learning models for multi-label classification, tackling a more realistic scenario in which people are not engaged in just one activity. Both use data from multiple sensors: Saeed et al. proposed a framework consisting of four-stream network architecture, which handles learning from each modality, and of a contextualization module that incorporates extracted representations to infer a user's context. Ehatisham-Ul-Haq et al. combined the classification of 6 primary activities of daily living (ADL), like sitting, walking, etc., with their association with 14 behavioral contexts (in a car, talking, indoor, outdoor, etc.). Similarly, Asim et al.[2] presented a framework for context-aware human activity recognition that uses a single sensor (phone accelerometer).

While all the previously cited works brought pioneer ideas, none of them regarded the temporal nature of the sensor data. Some like [4] model the temporal relation between dataset's samples, while

others treat each sample as standalone with the classical batch learning approach. No one kept in mind that sensor data are volatile, and they do not arrive all together but one at a time. A novel approach could be considering the data at hand as stream data and so applying online machine learning models.

For future real-world applications, it is important to tackle the problem using data collected in-the-wild, so that more realistic scenarios could be considered. To this end, we'll use the *ExtraSensory* dataset[3] collected in-the-wild. Moreover, as nothing can be found in literature about behavioral context recognition combined with online machine learning, we explore the feasibility of classifying primary ADLs first like sleeping and walking, keeping the more complex behavioral context (in a car, indoor, outdoor, etc.) classification for further development.

2 Implemented Strategy

The problem at hand regards the recognition of behavioral context working with sensor data as time series data. Hence online machine learning methods have been employed to carry out the classification of two primary ADLs, i.e. walking and sleeping.

The dataset used is The ExtraSensory Dataset, in which over 300k samples were collected in the wild from 60 users. Data came from everyday sensor devices (smartphones and smartwatches) and over 50 contextual labels are available.

2.1 The Dataset

The ExtraSensory dataset contains information from 60 users (also known as subjects or participants), each of whom is identified by a unique identifier (UUID). It has hundreds of instances from each user, often taken in one-minute intervals (but not necessarily in one long sequence, there are time gaps). Every example includes data from sensors (from the user's own smartphone as well as a smartwatch we provided). The user-reported context labels appear in the majority of cases.

2.1.1 Users

The users were mostly students (both undergraduate and graduate) and research assistants from the UCSD campus. 34 iPhone users, 26 Android users. 34 female, 26 male. 56 right handed, 2 left handed, 2 defined themselves as using both. Diverse ethnic backgrounds (each user defined their "ethnicity" how they liked), including Indian, Chinese, Mexican, Caucasian, Filipino, African American and more. Here are some more statistics over the 60 users:

	Range	**Average (standard deviation)**
Age (years)	18-42	24.7 (5.6)
Height (cm)	145-188	171 (9)
Weight (kg)	50-93	66 (11)
Body mass index (kg/mˆ2)	18-32	23 (3)
Labeled examples	685-9,706	5,139 (2,332)
Additional unlabeled examples	2-6,218	1,150 (1,246)
Average applied labels per example	1.1-9.7	3.8 (1.4)
Days of participation	2.9-28.1	7.6 (3.2)

2.2 Sensors

The sensors used were diverse and include high frequency motion reactive sensors. We report information about only those sensors we considered in our work (accelerometer, gyroscope, watch accelerometer). The following table specifies the different sensors, the format of their measurements for a single

example:

sensor	details	dimension	#us	#ex
accelerometer	Tri-axial direction and magnitude of acceleration. 40Hz for ∼20sec.	(∼800) x 3	60	308,306
gyroscope	Rate of rotation around phone's 3 axes. 40Hz for ∼20sec.	(∼800) x 3	57	291,883
watch accelerometer	Tri-axial acceleration from the watch. 25Hz for ∼20sec.	(∼500) x 3	58	282,527

2.2.1 Labels

As stated before, 50 contextual labels are available, from simple ADLs like walking, sleeping, shopping, cleaning, etc. to more complex context information like car passenger, in a meeting, etc. We have used following labels in our project:

Label	#users	#examples
SLEEPING	53	83055
FIX_walking	60	22136

Vaizman et al. have processed and cleaned the labels that were self-reported by users. Labels with prefix 'FIX_' are processed versions of original labels.

2.2.2 How was the data collected

Data was collected using the ExtraSensory mobile application. Vaizman et al. developed a version for iPhone and a version for Android, with a Pebble watch component that interfaces with both the iPhone and the Android versions. The app performs a 20-second "recording session" automatically every minute. In every recording session the app collects measurements from the phone's sensors and from the watch (if it is available), including: the phone's accelerometer, gyroscope (sampled in 40Hz), audio (sampled in 22kHz, then processed to MFCC feature representation), location, the watch's accelerometer (sampled in 25Hz) and compass and additional sensors if available (light, humidity, air pressure, temperature). The measurements from a recording session are bundled into a zip file and sent to the lab's web server (if WiFi is available, or stored on the phone until WiFi is available).

2.3 Dataset Exploration Analysis

A preliminary data exploration step has been carried out to first assess the composition of the dataset and to find relevant connections between sensor data and ADLs.

With the data at hand, we took into consideration a single label - among the two ones we wanted to analyze, i.e. walking and sleeping - to find out the time series' trends for each variable and the correlation between the single variable and the labels.

We started by analyzing the behavior we find heuristically easier to detect based on the features at hand, that is walking behavior. According to the work of Vaizman et al.[3], the most helpful features

to detect walking/not walking behavior are in order: phone gyroscope, phone acceleration and watch acceleration. During the analysis, we considered the comparison between walking and sleeping labels to better understand if the features at hand could be useful to distinguish between intuitively mutually exclusive labels. For example, watch's acceleration could be useful for distinguishing between WALKING vs SLEEPING behaviors since arms tend to move more while walking while not at all while sleeping.

The results of the above data analysis process will the discussed in the Results section.

2.4 Dataset Preparation

The next phase includes a set of operations to clean and create the dataset for model training.

With the complete and unaltered ExtraSensory dataset, we created 4 different datasets for WALKING/NON WALKING classification:

- a dataset with features from phone accelerator (mean magnitude, standard deviation, third and fourth moment, 1st,2bd and 3rd percentile, etc.) for a total of 27 features;
- a dataset with features from phone gyroscope, for a total of 26 features;
- a dataset with features from watch accelerometer, for a total of 46 features;
- a dataset with features from all the three sensors, for a total of 98 features.

At first, we chose to remove all rows with missing values in features columns. Of the complete dataset, 290 samples have missing values for phone accelerometer, 17,434 samples have missing values for gyroscope sensor, and 132,544 have missing values for watch accelerometer. All this samples have been discarded, for a total of 150,268 deleted samples, and 232,873 remaining samples.

Then for the label column, we set all nan values to 0. Specifically, for walking label all the 34,718 remaining samples with missing value in the walking label column have been set to 0. After this cleaning process, the remaining samples have this values distribution:

- False 217951
- True 14922

So the dataset is heavily unbalanced towards the not-walking class (false, i.e. 0).

The same steps have been applied for the creation of datasets for SLEEPING/NON SLEEPING classification, hence a total of 8 different datasets have been prepared. While cleaning the dataset for the sleeping class, a total of 150,268 sample with missing value in features columns have been deleted as before. Plus, a total of 48,016 samples with missing values in the sleeping label column have been set to 0, resulting in the following values distribution:

- False 185918
- True 46955

In this case too, the dataset is heavily unbalanced towards the not-sleeping class. This situation has been taken in consideration while evaluating the models results.

2.5 Training Phase

As we stated before, we chose to address the problem by adopting online machine learning models. The software used to implement these models is MOA.

For the sake of comparison with the previous work of Vaizman et al., we first trained a logistic regression classifier in online mode, by setting the MOA parameters as follows:

- learner: functions.SGD
- learning rate of default: 1.0E-4
- loss function: LOGLOSS
- instanceRandomSeed: 1
- sampleFrequency: 5000
- evaluation type: EvaluateInterleavedTestThenTrain
- evaluator: BasicAUCImbalancedPerformanceEvaluator

The most commonly reported model evaluation metric is the accuracy. This metric can be misleading when the data are imbalanced. Since we found that the dataset is imbalanced in both cases (sleeping and walking classes), we adopted the BasicAUCImbalancedPerformanceEvaluator to obtain meaningful evaluation metrics. Besides accuracy, we report Kappa and G-Mean values since these metrics take in consideration the imbalance in the dataset:

- Kappa statistic k - if the classifier is always correct, then $k = 1$; if its predictions coincide with the correct ones as often as those of a chance classifier, then $k = 0$.
- G-mean (geometric mean) - measures the balance between classification performances on both the majority and minority classes. A low G-Mean is an indication of a poor performance in the classification of the positive cases even if the negative cases are correctly classified as such.

We then adopted a Naive Bayes classifier with same evaluation method and sample frequency so that the results of both models could be compared.

Moreover, in order to provide a meaningful comparison between online learning and batch learning on this classification problems, we trained the same models in batch machine learning more, too. Specifically, we used the Weka platform to carry out these experiments with the following setups:

- Experiment Configuration Mode: Simple
- Experiment Type: train/test percentage split (randomize)
- Number of repetitions: 10

For the models' setup we set the same parameters as the ones used in online machine learning.

In the Results section we will discuss the results of both models.

3 Theory of models used

Theory of logistic regression classifier and Naive Bayes with online settings.

3.1 Logistic Regression

Logistic Regression (also called Logit Regression) is a supervised machine learning method dedicated to binary classification tasks. It is commonly used to predict the probability that an instance belongs to a particular class. If the estimated probability is greater than 50%, then the model predicts that the instance belongs to that class, and otherwise it predicts that it does not.

Just like the Linear Regression model from which it derives, a Logistic Regression model computes the weighted sum of the input features (plus a bias term), but instead of outputting the result directly like the Linear Regression model does, it outputs the logistic of this result:

$$\hat{p} = h_\theta(\mathbf{x}) = \sigma(\mathbf{x}^T\theta)$$

The objective of the training is to set the parameter vector θ so that its cost function, also called log loss, is minimized through the Gradient Descent optimization algorithm.

In an online learning process, the changes in the weights that happen at a given stage depend specifically on the (current) sample being presented and possibly on the current state of the model.

3.2 Bayesian Classification

:

Bayesian classification is based on Bayes' Theorem. Bayesian classifiers are the statistical classifiers.

Naive Bayes is a classification algorithm known for its low computational cost and simplicity. As an incremental algorithm, it is well suited for the data stream setting.

However, it assumes independence of the attributes, and that might not be the case in many real data streams.

It is based on Bayes' theorem, which may be stated informally as :

$$Posterior = \frac{Prior \times likehood}{evidence}$$

4 Results

In the following section, we'll discuss the results obtained from the steps described in the Implemented Strategy section, starting with the data exploration step.

4.1 Dataset Exploration

A stated before, Vaizman et al. found that the most useful features to detect walking behavior are phone gyroscope, phone acceleration, and watch acceleration. We explored the correlation between these features and the walking label by plotting time-series graphs where days of a week are the unit on the x-axis and the y-axis is for the variable that is being measured. Furthermore, we explored the features trends for the sleeping label too to assess the differences between two intuitively different labels. Figure 1 shows the graphs we plotted.

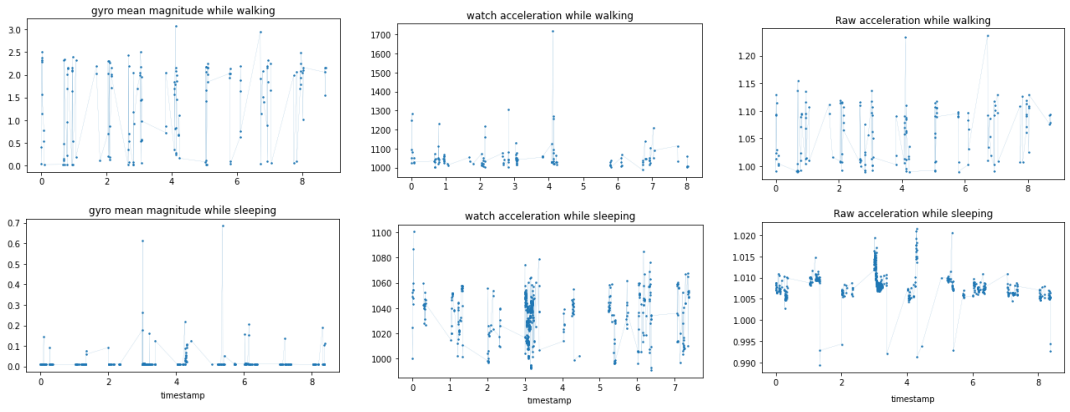

Figure 1: Graphs of different labels and features.

The gyro acceleration feature (mean magnitude) for WALKING takes values in a wider range [0.0-3.0] respect to the acceleration data for SLEEPING [0.0-0.7]. The same can be said for watch acceleration [1000-1700 vs 1000-1100]. Plus, the WALKING label data present more outliers than SLEEPING label data, demonstrating the fact that arms tend to move more while walking than sleeping and so watch acceleration could be a useful information for distinguishing between the two

behaviours. Phone acceleration while walking and sleeping doesn't seem to have any particular relevant trend: for sleeping label, data seems to collect around value 1.010 indicating the more stationary behaviour of phone while the user is sleeping. On the other hand, phone acceleration while walking ranges between [1.00-1.20] and mean value doesn't deviate too much from mean value of sleeping label (1.010).

4.2 Classification Results

Figure 4.2 reports the results we got from the test executed as described in the implemented strategy section.

Classifiers	Features	Labels	Accuracy	Kappa	G-Mean
Naive Bayes	Gyro Acceleration	Walking	0.83	0.50	0.71
	Raw Acceleration		0.83	0.48	0.70
	Watch Acceleration		0.80	0.51	0.72
	All Combined Features		0.84	0.52	0.73
	Gyro Acceleration	Sleeping	0.72	0.31	0.56
	Raw Acceleration		0.74	0.34	0.58
	Watch Acceleration		0.84	0.52	0.72
	All Combined Features		0.79	0.35	0.71
Logistic Regression	Gyro Acceleration	Walking	0.81	0.18	0.32
	Raw Acceleration		0.83	0.18	0.36
	Watch Acceleration		0.92	0.83	0.91
	All Combined Features		0.93	0.83	0.92
	Gyro Acceleration	Sleeping	0.81	0.29	0.60
	Raw Acceleration		0.83	0.26	0.54
	Watch Acceleration		1.00	0.99	0.99
	All Combined Features		0.99	0.99	0.99

As we can see in above table we find accuracy with different features and labels using two different classifiers. As stated before, the dataset is highly imbalanced, so other metrics as Kappa and G-Mean are more fit to depict the classifiers' results.

As we can see from the results, if we stopped at accuracy results it seems that all the 3 sensors' data are useful for distinguishing between walking/non walking behaviour. But, considering the more meaningful metrics of Kappa and G-means for imbalanced datasets, it's only the watch acceleration's features that are capable of distinguish between the two classes (k=0.83 and g-mean=0.91 with logistic regression classifier). The other two sensors' data are not helpful at all in this binary classification problem, contrasting with the Vaizman et al. findings that reported these two sensors as the best indicators for walking/not-walking behavior. With Naive Bayes the results obtained on this classification are not as high, only reaching a g-mean=0.73 and k=0.52 with all combined features.

As for the sleeping label, the highest performances have been reached with the watch acceleration's features again, with k=0.99 and g-mean=0.99 on logistic regression classifier. With Naive Bayes, too, it was the watch acceleration that scored the best results (k=0.52, g-mean=0.72). This is consistent with what Vaizman et al. reported in their work: among the tree sensors, the watch acceleration is the more helpful to distinguish sleeping/not-sleeping behavior. The results surpass the best one reported by Vaizman et al. for the sleeping behavior, that made use of the phone state data.

The second table 4.2 reports the results obtained by both classifiers with classical batch learning. In this case, balanced accuracy was considered ((TPR+TNR)/2). Overall, batch Naive Bayes beats the batch logistic regressor on both classification problems (BA=0.75 for walking and BA=0.74 for sleeping). But these results do not surpass the ones reached by the online logistic regressor, indicating that online machine learning classifiers could be a more useful tool for this kind of problem.

Classifiers	Features	Labels	Accuracy	Balanced Accuracy	TPR	TNR
Naive Bayes	Gyro Acceleration	Walking	0.89	0.74	0.91	0.57
	Raw Acceleration		0.90	0.73	0.92	0.53
	Watch Acceleration		0.83	0.73	0.84	0.61
	All Combined Features		0.87	0.75	0.89	0.61
	Gyro Acceleration	Sleeping	0.89	0.74	0.91	0.57
	Raw Acceleration		0.90	0.73	0.92	0.53
	Watch Acceleration		0.62	0.73	0.54	0.92
	All Combined Features		0.57	0.71	0.48	0.94
Logistic Regression	Gyro Acceleration	Walking	0.94	0.58	0.99	0.16
	Raw Acceleration		0.94	0.62	0.99	0.24
	Watch Acceleration		0.94	0.59	0.99	0.19
	All Combined Features		0.94	0.64	0.99	0.28
	Gyro Acceleration	Sleeping	0.80	0.50	1.00	0.00
	Raw Acceleration		0.80	0.53	0.99	0.06
	Watch Acceleration		0.82	0.63	0.95	0.31
	All Combined Features		0.85	0.72	0.93	0.50

5 Conclusion

The present work focuses on Naive Bayes and Logistic Regression for streaming data classification. As we tested the 2 different classifiers on The Extrasensory dataset, we observed that online Logistic Regressor reached best performances on both classification problems by using watch accelerator's data (k=0.99 and g-mean=0.99 for sleeping, and k=0.83 and g-mean=0.91 for walking). It has shown performances superior to the offline models with the same input features. In contrast, batch Naive Bayes has scored best results on both classification problems than the online counterpart. It would be interesting for future development to use the features that have been found most useful in state of art for sleeping classification - i.e. phone state - to assess if online models can achieve similar accuracy too.

References

[1] E. J. Kim, J. A. Jun, N.-S. Kim, and C. S. Pyo, "The method for hierarchical context awareness based on activity recognition," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 974–976, 2019.

[2] Y. Asim, M. A. Azam, M. Ehatisham-ul Haq, U. Naeem, and A. Khalid, "Context-aware human activity recognition (cahar) in-the-wild using smartphone accelerometer," *IEEE Sensors Journal*, vol. 20, no. 8, pp. 4361–4371, 2020.

[3] Y. Vaizman, K. Ellis, and G. Lanckriet, "Recognizing detailed human context in the wild from smartphones and smartwatches," *IEEE Pervasive Computing*, vol. 16, no. 4, pp. 62–74, 2017.

[4] A. Saeed, T. Ozcelebi, S. Trajanovski, and J. J. Lukkien, "End-to-end multi-modal behavioral context recognition in a real-life setting," in *2019 22th International Conference on Information Fusion (FUSION)*, pp. 1–8, 2019.

[5] M. Ehatisham-Ul-Haq, M. A. Azam, Y. Amin, and U. Naeem, "C2fhar: Coarse-to-fine human activity recognition with behavioral context modeling using smart inertial sensors," *IEEE Access*, vol. 8, pp. 7731–7747, 2020.